

PRZEMYSŁAW ŻARNECKI

---

PYTHON 02

## PYCHARM - SPOSÓB NA ZDOBYCIE LICENCJI:

The screenshot shows the PyCharm website homepage. At the top, there's a navigation bar with links: Developer Tools, Team Tools, Education (which is highlighted with a red box), Solutions, Support, Store, a search icon, and user account icons. Below the navigation is a banner for PyCharm, featuring a yellow and green abstract shape. The main content area has a section titled "Free License Programs" with two bullet points: one with a checkmark indicating renewal for students/teachers and one with a red X indicating it can't be shared. Below this is a large image of three cartoon characters (two adults and one child) looking at a computer screen. A call-to-action button labeled "Apply now" is overlaid on the image. To the right, there's a section for "Developer Tools" with a list of categories: Academic Licensing, Open Source, User Groups, Events Partnership, and Developer Recognition. A red arrow points from the "Education" link in the header to this section.

Developer Tools Team Tools Education Solutions Support Store

PyCharm

Coming in 2023.3 New UI What's New Features Learn Pricing Download

Free License Programs

- May be renewed free of charge as long as you are a student or a teacher.
- May not be shared with any third parties.

Academic Licensing Open Source User Groups Events Partnership Developer Recognition

Get free access to all developer tools from JetBrains!

Apply now

Options to get a free educational license

IntelliJ IDEA PyCharm RubyMine GoLand PHPStorm WebStorm CLion DataGrip Kotlinlang.org YouTrack

The screenshot shows the "Education" section of the JetBrains website. It features a central purple box for "JetBrains Academy" with a sub-section for "Full-fledged Professional or Free Community". To the left, there's a "FOR LEARNERS" section with "Programming languages" (with a note to select a language), "Career fields" (with a note to explore careers), and "Internships" (with a note to apply for internships). To the right, there's a "FOR EDUCATORS" section with "Teaching with JetBrains IDEs" (with a note to create courses), "Kotlin for education" (with a note to teach Kotlin courses), and "FOR TEAMS" (with a note to ensure up-to-date technical skills). A red box highlights the "FREE LICENSES" section under "FOR EDUCATORS", which includes options for students/teachers, educational institutions, and bootcamps. A red arrow points from the "Education" link in the header to this section.

Developer Tools Team Tools Education Solutions Support Store

JETBRAINS

FOR LEARNERS

Programming languages Select a language and try different approaches to learning it

Career fields Explore careers and see where programming could take you

Internships Apply for internships and flexible jobs for students

FOR EDUCATORS

Teaching with JetBrains IDEs Create courses and share your knowledge

Kotlin for education Teach a wide range of Kotlin courses

FOR TEAMS

Professional development Ensure your team has up-to-date technical skills

FREE LICENSES

For students and teachers JetBrains IDEs for individual academic use

For educational institutions JetBrains IDEs and team tools for classroom use

For bootcamps and courses JetBrains IDEs for your students

Full-fledged Professional or Free Community

JetBrains tools are FREE for education! Enjoy a professional developer experience using industry-leading tools to learn, teach, and collaborate.

Get started now

## W SUMIE NIECO PRZYPOMNIENIA

Nieczy  
i mówię  
sobie  
w jednej  
linii

```
punkty = 60
|
if punkty >= 65:
    print("Ocena pozytywna")
```

Yak cos
podniedne to
wiecie

## W SUMIE NIECO PRZYPOMNIENIA

punkty = 60

ZMIENNA

if punkty >= 65:

print("Ocena pozytywna")

punkty = int(input("Podaj ilość punktów"))

if punkty >= 65:

print("Ocena pozytywna")

## W SUMIE NIECO PRZYPOMNIENIA

punkty = 60

if punkty >= 65:

    print("Ocena pozytywna")

else:

    print("Ocena negatywna")

If ; else son  
sobie mówme.  
jedec tu mówme  
wcięć

## W SUMIE NIECO PRZYPOMNIENIA

```
if punkty >= 90:
```

```
    print("ocena 5")
```

```
elif punkty >= 80:
```

```
    print("ocena 4")
```

```
elif punkty >= 70:
```

```
    print("ocena 3")
```

```
elif punkty >= 65:
```

```
    print("ocena 2")
```

```
else:
```

```
    print("ocena 1")
```

## FUNKCJE ZAGNIEŻDŻONE - WARTO STOSOWAĆ. UWAGA NA WCIECIA

```
if warunek1:      # "zewnętrzna" instrukcja if  
    print("true")  jeśli będzie True wykona "2" if  
    if zagnieżdzony_warunek: # wewnętrzna instrukcja if  
        print("yes")  
    else:           # wewnętrzna instrukcja else  
        print("no")  
    else:           # zewnętrzna instrukcja else  
        print("false")  
  
Wykona jeśli 1  
będzie false
```

## FOR VS WHILE

	Pętla `for`	Pętla `while`
Wykorzystanie	Do iteracji po sekwencji	Do iteracji z warunkiem
Warunek stopu	Z góry określony (ilość iteracji)	Zależy od spełnienia warunku
Przykład	`for i in range(5):`	`while x < 10:`
Kontrola	Bardziej kontrolowana przez język	Może wymagać bardziej ręcznej kontroli
Czytelność	Zazwyczaj bardziej czytelna, gdy iterujemy po kolekcji	Czasami bardziej czytelna, gdy warunek jest złożony
Potencjalne nieskończone wykonanie	Bardziej bezpieczna, trudniej o nieskończoną pętlę	Potencjalnie bardziej ryzykowna, łatwiej o nieskończoną pętlę

## FOR - DEFINICJE

for element in sekwencja:

# Ciało pętli

# Wykonuje się dla każdego elementu w sekwencji

- element to zmienna tymczasowa, która przyjmuje wartość każdego elementu w sekwencji podczas iteracji.
- sekwencja to dowolny obiekt iterowalny, taki jak lista, krotka, napis, itp.:

lista = [1, 2, 3, 4, 5]

for element in lista:

    print(element)

Ten przykład wydrukuje liczby od 1 do 5, ponieważ pętla for iteruje przez elementy listy lista i wyświetla je na ekranie

Pętla for to jedne z podstawowych narzędzi do powtarzania operacji w python.

## FOR PRZYKŁADY

---

```
for i in range(1,11):
    print(i)
```

- 1
- 2 Po instrukcji for, Python wie, że ma wykonać następujący po niej kod, wielokrotnie.
- 3
- 4 1. Za każdym powtórzeniem, zmieniąca się zmieniła 'i', będzie zawierać inną liczbę. Zmieniąca się zmieniła 'i', oczywiście może się nazywać inaczej.
- 5
- 6 2. Wartości jakie będzie przyjmować zmieniąca się zmieniła 'i', będą w zakresie określonym przez funkcję range. Od 1 do 11.
- 7
- 8
- 9 3. Funkcja 'print', odwołuje się do zmiennej i, po czym wyświetla jej zawartość
- 10

## FOR PRZYKŁADY

---

```
lista = [1,2,'abc',3,4]  
  
for i in lista:  
    print(i)
```

---

```
1  
2  
abc  
3  
4
```

## FOR PRZYKŁADY

---

```
tekst = "Analityk"  
  
for ttt in tekst:  
    print(ttt)
```

A  
n  
a  
l  
i  
t  
y  
k

## WHILE - DEFINICJA

---

Pętla `while` w języku Python to struktura kontrolująca powtarzanie określonego bloku kodu tak długo, jak dany warunek jest spełniony. Jest to narzędzie programistyczne, które umożliwia iteracyjne wykonywanie operacji, z uwzględnieniem warunku zakończenia, który decyduje, kiedy pętla przestaje działać."

```
licznik = 0
```

```
while licznik < 5:
```

```
    print("Aktualna wartość licznika:", licznik)
```

```
    licznik += 1
```

zwiększenie zmiennej o 1 w każdym obiegu pętli

# WHILE

---

```
krok = 0  
  
while krok < 10:  
    print(krok)  
    krok = krok + 1
```

innu sposób zwiekszać

0  
1  
2  
3  
4  
5  
6  
7  
8  
9

---

# WHILE

---

```
while True:  
    print("hi")
```

hi  
hi  
hi  
hi  
hi  
hi  
hi  
hi  
hi  
hi

Co moimą?

- czekać na zdarzenia użytkownika
- sprawdzać czy pojawił się jakiś plik
- sprawdzać czy pojawiły się jakieś dane
- i wiele innych

# WHILE

---

```
lista = ['A', 1, 2, 'abc', 'x']

for element in lista:

    if element == 2:
        break
    else:
        print(element)

print ("Pętla została zakończona")
```

A  
1  
Pętla została zakończona

---

## WHILE - BREAK MOŻNA STOSOWAĆ TEŻ W FOR

---

```
lista = ['A', 1, 2, 'abc', 'x']

for element in lista:

    if element == 2:
        break
    else:
        print(element)

print ("Pętla została zakończona")
```

```
A
1
Pętla została zakończona
```

# DEFINIOWANIE FUNKCJI W PYTHON

---

```
def nazwa_funkcji(argumenty):  
    # Ciało funkcji  
  
    # Możesz wykonywać operacje tutaj  
  
    return wynik # Opcjonalnie zwracasz wynik
```

Gdzie:

- nazwa\_funkcji to nazwa twojej funkcji.
- argumenty to lista argumentów, które funkcja może przyjmować. Mogą być one opcjonalne.
- return to opcjonalny operator, który pozwala na zwrócenie wartości z funkcji.

# DEFINIOWANIE FUNKCJI W PYTHON

---

Przykład definicji i użycia prostej funkcji w Pythonie:

```
def witaj(imie):  
    powitanie = "Witaj, " + imie + "!"  
  
    return powitanie  
  
# Wywołanie funkcji i wydrukowanie wyniku  
  
print(witaj("Jan")) # Wynik: "Witaj, Jan!"
```

# DEFINIOWANIE FUNKCJI W PYTHON

---

```
def say_hello():
```

```
    # Blok należący do funkcji.
```

```
    print('hello world')
```

```
    # Koniec funkcji.
```

```
say_hello() # Wywołanie funkcji.
```

```
say_hello() # Ponowne wywołanie funkcji.
```